commodore

TOOL 64

TOOL 64

VERSIONE ITALIANA

MANUALE PER L'UTILIZZATORE

COD. 6022

COMMENTI E SEGNALAZIONI DI ERRORI PER IL LETTORE

Questo manuale è stato accuratamente riveduto e corretto prima della stampa. Se tuttavia doveste riscontrare qualche errore vi saremmo grati se vorreste comunicarcelo. Sono inoltre graditi eventuali commenti, critiche e suggerimenti.
Augurandovi un buon lavoro con TOOL 64 porgiamo i migliori saluti.

Rinaldo Farina

R. Farina
Software Manager

Commodore Italiana S.p.A.
Via F.lli Gracchi, 48
20092 Cinisello Balsamo (MI)

Commodore Italiana SpA

Via F.lli Gracchi, 48 - 20092 Cinisello Balsamo
Tel. 02/618321

6022 - 3/84 - 2M

Il software "THE TOOL" per il personal computer COMMODORE 64 è un potente complemento al vostro sistema che rende più semplice l'uso del computer e nello stesso tempo ne aumenta notevolmente le possibilità.

Il software consiste in un nuovo set di istruzioni tipo BASIC, di facile e rapida comprensione, che migliorano la gestione dello schermo, permettono di tracciare linee verticali ed orizzontali, di ottenere efficienti routine per l'acquisizione dati, di variare i colori presenti sullo schermo, di disegnare in grafica ad alta risoluzione con la contemporanea presenza di testo ed infine di memorizzare su floppy disk e richiamare dallo stesso pagine di schermo con grafica e testo.

Il software "THE TOOL" comprende inoltre alcune istruzioni di estrema utilità per la programmazione nella stesura e nel debug dei programmi, nonchè altre caratteristiche interessanti quali una funzione riservata al joystick e delle istruzioni per gestire stringhe di caratteri.

Le istruzioni per il supporto DOS sono state inserite per gestire con maggior semplicità l'unità a floppy disk.

Questo manuale è stato redatto allo scopo di costituire sia uno strumento per imparare ad usare THE TOOL, che uno strumento di riferimento. Si consiglia di leggere attentamente il manuale stesso per ottenere tutti gli elementi che assicurano un'utilizzazione efficace e razionale del software "THE TOOL".

Un'occhiata agli esempi e la loro lettura mentre girano sul vostro COMMODORE 64 vi permetteranno di prendere molto rapidamente confidenza con tutte le possibilità del sistema.

GENERALITÀ

Il software THE TOOL è memorizzato su una cartuccia.

È sufficiente inserire la cartuccia nel connettore di espansione della memoria, collocato sul retro del COMMODORE 64, ed accendere quest'ultimo. THE TOOL viene automaticamente caricato nella memoria del computer, e vi sono ancora 30 k della stessa a vostra disposizione. In questo momento tutta la potenza del "THE TOOL" è ai vostri ordini.

Nota: la forma della cartuccia dovrebbe evitare ogni confusione per il suo inserimento. In caso di dubbio, fate riferimento alla "Guida per l'utente" del COMMODORE 64 che vi permetterà di individuare con sicurezza il connettore in cui inserire la cartuccia.

GENERALITÀ

Il generatore di schermo presente nel software "THE TOOL" ha lo scopo di facilitare l'acquisizione dati sullo schermo. L'acquisizione, infatti, diventa più affidabile che con le normali routine in BASIC e l'esecuzione, fondamentale nella gestione dello schermo, diventa più veloce essendo il software scritto in linguaggio macchina.

THE TOOL controlla le seguenti funzioni:

comandi di visualizzazione per tracciare linee verticali ed orizzontali, per effettuare lo scroll dello schermo, etc...

comandi di acquisizione per definire zone di acquisizione e controlli associati, per allocare contenuti di particolari zone dello schermo a variabili BASIC, per un più completo utilizzo delle possibilità di stampa.

comandi per la gestione dello schermo a pagina per caricare su floppy e richiamare da floppy pagine di schermo.

Nelle pagine seguenti, i nomi utilizzati per le variabili sono arbitrari e sono scelti in modo da ottenere la miglior comprensione. Qualunque variabile BASIC può essere scelta a tale scopo, tranne le variabili riservate per THE TOOL: "zo" ed "ok".

Lo schermo viene definito come una superficie su cui disegnare con l'origine degli assi nell'angolo in alto a sinistra. L'origine ha coordinate 1.1. Ogni punto dello schermo è definito per mezzo delle coordinate:

l = numero di riga
c = numero di colonna

GENERALITÀ (continua)

In generale si ha:

l : numero di linea (o di riga) (1 <= 1 <= 25)
c : numero di colonna (1 <= c <= 40)
ln: lunghezza di riga (o numero delle colonne) - linea orizzontale
lg: altezza di colonna (o numero delle righe) - linea verticale

Per una miglior comprensione di quanto detto in questo capitolo, provate questi comandi sullo schermo del vostro sistema, in modo diretto o da programma.

Una pagina di schermo è qualunque cosa si trova sullo schermo in un determinato istante. È possibile creare pagine complete sullo schermo, caricarle su floppy disk, richiamarle direttamente sullo schermo, inserire dei dati ed effettuare anche altre operazioni.

MODO GRAFICO/MODO TESTO

Si vedrà che è possibile utilizzare contemporaneamente testo e grafica e memorizzare il tutto su floppy disk. Le istruzioni grafiche saranno esaminate nel prossimo capitolo. Per ora le istruzioni che seguono non verranno utilizzate con la grafica.

Esistono due modi di funzionamento: il modo TEXT (per i testi) ed il modo GRAPHIC (per la grafica). Nel capitolo 3 saranno esaminate le istruzioni in modo Text, mentre nel capitolo 4, quelle per il modo Graphic. Il modo di default è comunque il modo Text.

tline

Funzione: tline permette di disegnare sullo schermo una linea orizzontale definita per mezzo della lunghezza e dell'origine.

Sintassi: tline ln, l, c

dove
ln: lunghezza della linea orizzontale
l : numero di riga dell'origine della linea orizzontale
c : numero di colonna dell'origine della linea orizzontale

Utilizzazione: il comando tline permette di disegnare tabelle e cornici sullo schermo o di sottolineare dati e titoli.

Esempio: tline 22, 1, 1
tline 22, 12, 1

Queste due istruzioni tracciano due linee orizzontali parallele nella parte superiore dello schermo.

tcol

Funzione: tcol permette di disegnare sullo schermo una linea verticale definita per mezzo della lunghezza e dell'origine.

Sintassi: tcol, ln, l, c

dove
ln: lunghezza della linea verticale
l : numero di riga dell'origine della linea verticale
c : numero di colonna dell'origine della linea verticale

Utilizzazione: il comando tcol permette di disegnare tabelle e cornici sullo schermo (assieme al comando tline).

Esempio:

```
tcol 12, 1, 1
tcol 12, 1, 22
```

Queste due istruzioni tracciano due linee verticali parallele nella parte superiore dello schermo.

Note:

```
10 tline 22, 1, 1 : tline 22, 12, 1
20 tcol 12, 1, 1 : tcol 12, 1, 22
```

Queste due linee di comandi BASIC tracciano una cornice sullo schermo. Notare che le intersezioni agli angoli sono disegnate correttamente.

clear

Funzione: clear cancella una zona definita sullo schermo.

Sintassi: clear 1, c, lg, ln

dove
l : numero di riga del punto di partenza
c : numero di colonna del punto di partenza
lg: altezza della zona da cancellare (numero di righe)
ln: lunghezza della zona da cancellare (numero di colonne).

Utilizzazione: il comando clear permette di cancellare una parte dello schermo definita per mezzo dei parametri.

Esempi: per cancellare la prima riga dello schermo

clear 1, 1, 1, 40

per cancellare la cornice definita negli esempi precedenti e tutto ciò che è compreso in essa

30 clear 1, 1, 12, 22

(cancella una "finestra" di 12 righe per 32 colonne).

out

Funzione: visualizzazione sullo schermo una stringa a$ in una posizione data.

Sintassi: out a$, l, c

dove
a$: una stringa qualunque
l : numero di riga del punto di partenza
c : numero di colonna del punto di partenza

Utilizzazione: il comando out visualizza una qualunque stringa in un qualunque punto dello schermo senza dover usare il cursore. Out rende inoltre più facile il riempimento di maschere.

Esempio:

```
40 a$ = "JOHN SMITH"
50 out a$, 5, 10
```

Questa sequenza di comandi visualizza la scritta JOHN SMITH sulla quinta riga dall'alto a partire dalla decima colonna.

rev

Funzione: rev trasforma in modo inverso una porzione di schermo definita dal suo punto di origine, dalla lunghezza, dall'altezza e dal colore.

Sintassi: rev l, c, lg, ln [,co]

dove
l : numero di riga del punto di partenza
c : numero di colonna del punto di partenza
lg : altezza della finestra (o numero delle righe)
ln : lunghezze della finestra (o numero delle colonne)
co: nuovo colore: opzionale (da 0 a 15 - vedi pag. 3-21)

Utilizzazione: il comando rev permette di operare su una porzione dello schermo. Usando l'istruzione più di una volta è possibile far lampeggiare la finestra. La frequenza del lampeggio può essere regolata per mezzo di loop temporali (for: next).

Esempio: utilizzare il modo inverso p-er i nomi dell'esempio precedente.

140 rev 5, 1, 1, 10, 3 : rev 6, 10, 1, 11, 4

scroll

Funzione: scroll provoca uno scorrimento verso l'alto o verso il basso, verso destra o verso sinistra di una porzione dello schermo o di una finestra definita dal suo punto di origine, dalla lunghezza e dall'altezza.

Sintassi: scroll l, c, lg, ln, ty

dove
l : numero di riga del punto di partenza
c : numero di colonna del punto di partenza
lg : altezza della finestra o numero delle righe
ln : lunghezza della finestra o numero delle colonne
ty : tipo dello scrolling con:
–n: verso l'alto (up)
–d: verso il basso (down)
l : verso sinistra (left)
r : verso destra (right)

Utilizzazione: il comando scroll permette di visualizzare file di dati troppo grandi per essere visualizzati una sola riga. Con un semplice codice è possibile quindi far scorrere i dati in una finestra e, non appena si trova il dato desiderato, memorizzarlo, procedendo quindi con l'esecuzione del programma.

Esempio:

```
10 sclear
30 a$ = "JOHN" : b$ = "PATTY" : c$ = "JULIE"
40 out a$, 11, 4 : out b$, 11, 10 : out c$, 11, 17
50 for i = 0 to 9
60 scroll 10 - i, 4, 2, 4, u
70 clear 11 - i, 4, 1, 4,
80 scroll 11 + i, 17, 2, 5, d
90 clear 11 + i, 17, 1, 5
100 for j = 1 to 200 : next j
110 next i
120 end
```

Fate girare il programma ed osservate che cosa succede. Quindi eliminate le linee 70 e 90 e notate le differenze. effettuate poi esempi simili con lo scrolling a sinistra ed a destra.

Nota: durante lo scroll non viene cancellato ciò che era visualizzato in precedenza.

GENERALITÀ

Le istruzioni di acquisizione di THE TOOL hanno lo scopo di essere utilizzate in sostituzione delle istruzioni INPUT e GET, per permettere un'acquisizione dati omogenea, controllata, efficiente ed affidabile.

Questa sezione è dedicata ai comandi per l'acquisizione. Una zona dello schermo viene definita per mezzo di alcuni parametri (punto di origine, dimensioni, tipo permesso di caratteri, possibilità uscita (oltre che return) ed un numero di identificazione). (**decz**)

Una volta definita la zona, il comando **reqz** istruisce THE TOOL a prelevare i dati da essa, effettuare i controlli desiderati e riprodurre le funzioni di controllo della pagina come: spostare il cursore, cancellare, inserire. Il comando **inz** effettua il trasferimento dell'informazione da una zona ad una stringa (il comando **outz** effettua l'operazione inversa).
Su una pagina di schermo possono essere definite fino a 128 differenti zone.

```
input a$    decz 1    : definizione della zona
            reqz 1    : acquisizione controllata
            inz  1, a$: trasferimento in a$
```

Ulteriori informazioni saranno date nelle pagine seguenti.

decz

Funzione: decz permette di definire tutti i parametri relativi ad una zona dello schermo per controllare l'acquizione dati.

Sintassi: decz n, l, c, ln [, ty] [, f, $]

dove
n : numero di identificazione (o < = n < = 127)
l : numero di riga del punto di partenza (1 < = l < = 23)
c : numero di colonna del punto di partenza (1 < = c < = 22)
ln: lunghezza totale della zona (1 < = ln < = 255)
ty: tipo della zona di controllo con:
— n : solo dati numerici
— m: dati in lettere maiuscole
— r : altre possibilità di uscita oltre a RETURN
* per una zona numerica: quasi tutti i tasti tranne quelli numerici
* per una zona alfanumerica: tutti i tasti non alfanumerici.

Il codice ASCII dei tasti premuti per uscire viene posto nella variabile zo, per cui è possibile definire ed usare i tasti di controllo nel seguente modo:
da F1 ad F8: zo = da 133 a 140
CRSR/DOWN: zo = 17
CRSR/UP: zo = 145
SHIFT/RETURN: zo = 141
CTRL/da A a Z: zo = da 1 a 26

— p: zona definita in stampa + f$

f$: stringa contenente il formato di stampa.

Un formato può definire una zona. Quando si prendono o si visualizzano dati in questa zona, il suo contenuto è formattato. Il formato è memorizzato in una stringa (in questo caso f$). I caratteri di controllo sono:

— 9: numerico
— 8: numerico; poni a zero se zero
— 1: posizione del segno (solo −)
— 2: posizione del segno (con + o −)

Qualunque carattere alfanumerico può essere inserito in un format.

Esempio: f$ = "$ 19,999,998.88"

Utilizzazione: decz permette di definire la zona di acquisizione con la massima precisione possibile. È possibile operare con efficienza e facilità in una zona utilizzando semplicemente il suo numero di identificazione.

decz

Esempio: decz 2,1,2,10,n,p,"199 999.88"

Questo esempio definisce una zona (numero 2) con un punto di origine (1,2) lunga 10 caratteri. Sono accettati solo valori numerici (n). Il formato di stampa provoca la stampa del solo segno - se è meno ed uno zero viene forzato al termine del numero. Il numero sarà di 5 cifre con due cifre dopo il punto decimale.

Note: Nella stessa istruzione decz può essere combinato più di un tipo. Sono accettate tutte le seguenti combinazioni:

n	: solo numeri	– RETURN per lasciare la zona
n,r	: solo numeri	– un tasto programmabile per lasciare la zona
n,p	: solo numeri formattati	– RETURN per lasciare la zona
n,r,p	: solo numeri formattati	– un tasto programmabile per lasciare la zona
m	: solo lettere maiuscole	– RETURN per lasciare la zona
m,r	: solo lettere maiuscole	– un tasto programmabile per lasciare la zona
r	: qualunque carattere è accettato	– un tasto programmabile per lasciare la zona
p	: (equivalente ad n,p)	
r,p	: (equivalente ad n,r,p)	
(nulla)	: qualunque carattere è accettato	– RETURN per lasciare la zona

Se i dati vanno in overflow sulla zona di acquisizione, la zona viene illuminata e quindi cancellata totalmente. Il cursore si riporta all'inizio della zona. Ciò impedisce di lasciare la zona se il formato non è corretto (il formato viene controllato premendo RETURN).

Se provate ad usare l'istruzione decz senza usare una istruzione reqz subito dopo, non accade nulla, ma la dichiarazione rimane.

reqz

Funzione: reqz effettua l'acquisizione secondo i parametri specificati nella precedente istruzione decz.

Sintassi: reqz n

dove
n: numero di identificazione della zona

Utilizzazione: il comando reqz viene usato in coppia con il comando decz già descritto. Se non esiste una corrispondente istruzione decz (oppure se la stessa non è stata caricata in una pagina di schermo), viene generato un messaggio di errore.

Con questo comando, THE TOOL effettua l'acquisizione dati sotto il suo controllo, come definito nella dichiarazione di zona. Durante l'esecuzione del comando, il cursore viene portato all'inizio della zona interessata.

Nella zona possono essere usati i comandi CLR, HOME, CURSOR BACK e FORWARD, INST/DEL.

L'acquisizione dati è controllata come definita nella dichiarazione di zona.

Se il modo di definizione della zona è r ed il codice di uscita è differente da 13 (zo), il cursore ritorna nella posizione che occupava durante l'uscita dalla zona quando si effettua un nuovo ingresso nella stessa.

Esempio: se avete già introdotto l'esempio precedente (decz 2, ...) introducete ora:

reqz 2

e notate quello che succede. Saranno accettati solo valori numerici.

inz

Funzione: inz trasferisce il contenuto di una zona in una variabile di stringa.

Sintassi: inz n, a$

dove
n : numero di identificazione della zona
a$: una qualunque variabile di stringa

Utilizzazione: il comando inz è utilizzato per trasferire il contenuto di una zona in una stringa, quando tutti i dati in quella zona sono corretti.

Esempio: inz 2, a$

Dopo aver effettuato i due esempi precedenti, questo comando trasferisce il contenuto della zona in a$. Per verificare se è vero, battere:

out a$,3,5

```
10 decz 1,25,10,5,r
20 reqz 1 : inz 1, a$
30 print zo, a$
40 goto 20
```

Note: la variabile di stringa che riceve il contenuto di una zona deve avere la stessa lunghezza di quella zona, indipendentemente dal contenuto.

outz

Funzione: outz visualizza una stringa in una zona

Sintassi: outz n, a$

dove
n : numero di identificazione della zona
a$: qualunque variabile di stringa

Utilizzazione: il comando outz è l'istruzione opposta di inz. I dati vengono trasferiti da una variabile ad una zona. Questa istruzione è simile all'istruzione **out**, ma è più facile da usare e più precisa.

Più facile da usare perché è sufficiente il numero di identificazione della zona anziché le coordinate del punto iniziale.

Più precisa perché assume tutti i controlli definiti nella zona con il comando decz.

Esempio: usando ancora gli esempi precedenti, battere a$ = "11111"
outz = 2,a$

Note: se nella zona definita per la stampa si verifica un'overflow dei decimali, il numero viene troncato. Se si verifica invece un'overflow di capacità, la zona viene riempita con "******";

nel caso di un comando outz in una zona dichiarata senza impegno (stampa senza 1 o 2), di una stringa con un +, un −, un blank come primo carattere, la stringa può essere troncata di una cifra, oppure sullo schermo può essere sostituita con degli asterischi. Per evitare ciò, occorre estrarre il segno con la sequenza:

a = len (mo$) − 1 : mo$ = right$ (mo$,a)

clearz

Funzione: clearz cancella una zona.

Sintassi: clearz nl [to n2]

dove
n1, n2: numeri di identificazione di una zona.

Utilizzazione: il comando clearz può essere usato per cancellare i contenuti di una maschera senza cancellare la maschera stessa, dopo un'acquisizione e prima di una nuova acquisizione.

Esempio: clearz 2

carget

Funzione: carget arresta l'esecuzione di un programma ed attende l'introduzione di un carattere predefinito.

Sintassi: carget a$ [, 1, c]

dove
a$: stringa contenente i caratteri accettati
l,c: (opzionali) posizione dove deve apparire il cursore sullo schermo

Utilizzazione: il comando carget sostituisce la ben nota sequenza in BASIC:

100 get a$: if a$ = " " then 100

con ulteriori possibilità

100 carget "abc"

arresta l'esecuzione del programma fino a che non vengono introdotti a, b oppure c.

Dopo un comando carget, le variabili ok e zo assumono i seguenti valori:
ok: puntatore di posizione del carattere scelto nella stringa (a ok=1, b ok=2, c ok=3)
zo: codice ASCII del carattere scelto (a zo=65, b zo=66, c zo=67)

Esempio: BASIC

```
100 get a$ : if a$= " " then 100
110 if a$= "A" then gosub 1000: goto 200
120 if a$= "B" then gosub 2000: goto 200
130 if a$= "C" then gosub 3000: goto 200
140 if a$= "Z" the gosub 4000: goto 200
150 goto 100
200 rem *** fine del programma ***
```

THE TOOL

```
100 a$= "ABCZ" : carget a$, 24, 20
110 on ok then gosub 1000, 2000, 3000, 4000
200 rem *** fine del programma ***
```

È evidente il risparmio di linee di programma.

Note: Se a$ è una stringa vuota (a$= " "), tutti i caratteri sono accettati.

100 carget " "

Questa istruzione arresta l'esecuzione del programma finché non viene premuto un tasto qualunque.

Usando il comando carget con i parametri 1, c'è possibile definire la frequenza di lampeggiamento del cursore con l'istruzione poke 703,n dove n : valore della frequenza del lampeggio (normale n=100).

GENERALITÀ

In applicazioni commerciali, le maschere di acquisizione dati sono spesso simili e vengono utilizzate più di una volta con differenti informazioni. In un normale programma BASIC, una nuova maschera deve essere ricreata ogni volta con notevole dispendio sia di tempo che di memoria.

THE TOOL definisce una pagina dello schermo con tutte le istruzioni di dichiarazioni (decz) e la memorizza su floppy disk evitando la necessità di creare nuovamente la maschera. Applicazioni differenti possono quindi avere le proprie pagine di schermo con le proprie maschere. La codifica del programma è notevolmente semplificata.

La pagina può essere successivamente richiamata sullo schermo ed in essa si possono inserire i dati. È sempre possibile scambiare la maschera presente sullo schermo con una memorizzata su floppy, ad esempio un maschera di HELP.

Se, durante un'acquisizione, si dimentica ad esempio la sintassi di un comando di controllo, il programma può facilmente effettuare uno scambio di pagine richiamando sullo schermo una pagina di HELP e salvando la pagina di lavoro sul floppy. Dopo aver effettuato le verifiche necessarie, si effettua l'operazione inversa e l'esecuzione del programma riprende dal punto in cui era rimasto. Il cursore riappare nel punto esatto dove si trovava prima di effettuare lo scambio.

Le tre istruzioni descritte in questa sezione effettuano le seguenti azioni:

- ssave : memorizza su floppy la pagina presente sullo schermo
- sload :richiama sullo schermo una pagina memorizzata su floppy
- sclear: cancella lo schermo

Note: come si vedrà con più dettagli nel capitolo seguente, le istruzioni ssave ed sload possono essere usate sia in modo Text che in modo Graphic.
Una pagina di testo sullo schermo può occupare 8-9 blocchi in funzione del numero delle zone.
Una pagina grafica occupa invece 33 blocchi.
È possibile caricare una pagina che non corrisponde al modo corrente (es. caricare una pagina grafica in modo Text). La pagina specificata verrà comunque caricata nella corretta parte di memoria.

ssave

Funzione: ssave memorizza la pagina dello schermo su floppy disk

Sintassi: ssave du, "nome" [, n1] [, n2 to n3] [, ...]

dove
du : numero logico dell'unità a floppy disk (normalmente 8)
nome : nome assegnato alla pagina
n1, n2, n3: zone associate (opzionali)

Utilizzazione: il comando ssave salva la pagina presente sullo schermo. Anche le zone (decz) possono essere salvate per mezzo dei parametri n1, n2 ed n3 che sono opzionali.
In ogni caso, viene salvata l'intera pagina (maschera, dati, etc.)

Esempio: ssave 8, "prgl", 1, 12, 30 to 70

Note:

1) ssave 8, "nome", 0 to 127 salva tutte le dichiarazioni associate. Non ci sono errori se sono salvate le zone non dichiarate.
2) può anche essere usata la forma ssave 8, " @ 0 : prgl"
3) ssave memorizza l'intero schermo (testo o grafica). Nel caso di modo grafico, le zone associate danno un errore di sintassi.

sload

Funzione: sload richiama sullo schermo una pagina precedentemente memorizzata su floppy disk.

Sintassi: sload du, "nome"

dove
du : numero logico dell'unità a floppy disk (generalmente 8)
nome: nome assegnato alla pagina

Utilizzazione: il comando sload è l'istruzione inversa di ssave. La pagina richiamata viene direttamente visualizzata sullo schermo. Tutte le zone vengono nuovamente dichiarate.

Esempio: sload 8, "progl"

Note: nessuna

sclear

Funzione: sclear cancella completamente una pagina dello schermo

Sintassi: sclear

senza parametri

Utilizzazione: il comando sclear cancella totalmente lo schermo, compresi i dati acquisiti con le istruzioni decz, regz, ed inz. Tuttavia il comando non distrugge le dichiarazioni dei dati.

Esempio: sclear

Note: nessuna

screen

Funzione: screen permette di modificare i colori dello schermo, dei bordi e dei caratteri visualizzati con THE TOOL.

Sintassi: screen sc, br [, cr]

dove
sc: colore dello schermo (0 <= sc <= 15)
br: colore dei bordi (0 <= br <= 15)
cr: colore dei caratteri (0 <= cr <= 15) (parametro opzionale)

Utilizzazione: con il comando screen è possibile definire facilmente i colori dello schermo, dei bordi e dei caratteri visualizzati con THE TOOL (per mezzo dei comandi tline, tcol, out, outz, regz - fare riferimento alle singole istruzioni).

Se il colore dei caratteri è ad esempio BLU, la modifica di questo colore con il comando screen avrà effetto solo su quei caratteri visualizzati per mezzo delle istruzioni di THE TOOL.

Esempi:

```
10 rem *** DISEGNAMO ALCUNE BANDIERE ***
20 sclear : screen 1,6,2 : out "U.S.A", 12,20 : carget" "
30 sclear : screen 1,5,2 : out "ITALIA", 12,20 : carget" "
40 sclear : screen 0,2,7 : out "GERMANIA", 11,7 : carget" "
50 sclear : screen 1,2,2 : out "GIAPPONE", 11,9 : carget" "
```

Note: 1) Tavola dei codici colore

Parametri	Colore	Codice
	Nero	0
	Bianco	1
	Rosso	2
	Ciano	3
validi	Porpora	4
per	Verde	5
sc, br	Blu	6
e	Giallo	7
cr	Arancione	8
	Arancione chiaro	9
	Rosa	10
	Ciano chiaro	11
	Porpora chiaro	12
	Verde chiaro	13
	Blu chiaro	14
	Giallo chiaro	15

2) I comandi STOP e RESTORE riportano i colori originali (schermo nero e caratteri bianchi).

GENERALITÀ

Nel precedente capitolo è stata introdotta la distinzione tra modo Text e modo Graphic. in esso è stato descritto il generatore di schermo con le istruzioni di visualizzazione, acquisizione dati e gestione delle pagine di schermo che funzionano sullo schermo normale 40 x 25 corrispondente al modo Text nel software THE TOOL.

Vediamo ora le istruzioni che permettono di disegnare su schermo grafico ad alta risoluzione (320 x 200). Lo schermo grafico viene definito come una superficie su cui disegnare con l'origine degli assi nell'angolo in basso a sinistra. ogni punto dello schermo grafico ad alta risoluzione è definito per mezzo di una coppia di coordinate (ascissa ed ordinata).

graphic

Funzione: graphic pone il COMMODORE 64 in modo Graphic ad alta risoluzione.

Sintassi: graphic

senza parametri

Utilizzazione: il comando graphic deve essere usato quando sta per essere eseguita una sequenza di istruzioni grafiche. Il comando riserva la memoria necessaria per la gestione della grafica ad elevata risoluzione.

Esempio:

```
10 tline 40,1,1 : tline 40,25,1
20 tcol 25,1,1 : tcol 25,1,40
30 ssave 8, "frame"
40 graphic
50 rem *** serie di istruzioni grafiche (fare riferimento alle pagine successive) ***
60 ssave 8, "graphic"
70 rem *** siamo ancora in modo graphic ***
```

Note: Quando si desidera ritornare in modo Text è sufficiente usare l'istruzione text (pagina 4 - 7).

move

Funzione: move permette di spostare il cursore grafico in qualunque parte dello schermo.

Sintassi: move x, y

dove
x: coordinata orizzontale (0 <= x <= 319)
y: coordinata verticale (0 <= y <= 199)

Utilizzazione: il comando move permette di disegnare un punto od una linea in qualunque posizione dello schermo. Il comando move sposta solamente il cursore, senza disegnare nulla sullo schermo.

Esempio:

```
10 graphic
20 move 0,0 : rem *** sposta il cursore nell'origine (angolo in basso a sinistra) ***
30 i = 160 : j = 100 : move i, j
40 text
50 rem *** esempi più completi nelle pagine che seguono ***
60 ...
```

Note: il punto origine in modo Graphic è differente per motivi pratici da quello del generatore di schermo. Infatti, la visualizzazione e l'acquisizione sono normalmente effettuate dall'alto verso il basso per cui l'origine in modo Text è nell'angolo superiore a sinistra. Invece, quando si disegna, il punto origine si trova in basso a sinistra.

draw

Funzione: draw disegna (o cancella) una linea dal punto dove si trova il cursore (dopo averlo spostato ad esempio con un comando move) al punto di coordinate x, y.

Sintassi: draw x, y, ty

dove
x : coordinata orizzontale (0 <= x <= 319)
y : coordinata verticale (0 <= y <= 199)
ty: una variabile numerica
  1: traccia una linea
  0: cancella una linea

Utilizzazione: il comando draw viene utilizzato ogni qualvolta occorra tracciare o cancellare una linea retta sullo schermo. in unione al comando move è possibile ottenere qualunque tipo di disegno con linee rette.

Esempio:

```
05 rem *** disegna un cubo in 3D ***
10 graphic : sclear
20 a=0 : b=20 : C=40 : d=100
25 move a, a : rem *** cursore nell'origine (angolo in basso a sinistra) ***
30 draw a,d,1 : draw d,d,1 : draw d,a,1 : draw a,a,1
40 draw c,b,1 : draw c+d,b,1 : draw c+d, b+d,1
50 draw c,b+d,1 : draw a,d,1
60 move d,a : draw d+c,b,1
65 move c,b : draw c,b+d,1
70 move d,d : draw d+c, d+b,1
80 carget " " : rem *** premere qualunque tasto per ripartire ***
90 next
```

Note: 1) Aggiungendo al precedente esempio le linee 130, 140, 150, 160, 165 e 170 come da 30 a 70, ma con 0 anzichè 1, si cancella il cubo, linea dopo linea. L'istruzione sclear ha lo stesso effetto in modo più rapido (vedere pag. 3 - 20).

2) Il comando carget (pag. 3 - 16) senza memoria di riga e di colonna può essere usato anche in modo grafico.

3) Si sarebbe anche potuto usare:

draw a,d,dr

dove
dr = 1 l'istruzione disegna
dr = 0 l'istruzione cancella

plot

Funzione: plot disegna o cancella un punto di coordinate x,y

Sintassi: plot x,y,ty

dove
x : coordinata orizzontale (0 <= x <= 319)
y : coordinata verticale (0 <= y <= 199)
ty: una variabile numerica
1: disegna un punto
0: cancella un punto

Utilizzazione: il comando draw è usato per disegnare o cancellare una cosa diversa da una linea retta.

Esempio:

```
05 rem *** disegna un cerchio in un quadrato ***
10 graphic
20 a=0 : d=100 : e=50 : pi=3.14159 : pr=.01 : dr=1
25 move a,a : rem *** cursore nell'origine (angolo in basso a sinistra) ***
30 draw a,d,dr : draw d,d,dr : draw d,a,dr : draw a,a,dr
40 for i=0 to 2*pi step pr
50 plot e*(1+cos(i)), e*(1+sin(i)), dr
60 next i
80 carget " " : rem *** premere qualunque tasto per ripartire ***
90 text
```

Note: facendo girare lo stesso programma con dr=0 (linea 20) viene cancellato il circolo e la cornice. Il disegno rimane in memoria finchè non viene usato il comando sclear o draw di tipo 0.

point

Funzione: point verifica se un punto è o non è disegnato sullo schermo alle coordinate x, y.

Sintassi: point x, y, ty

dove
x : coordinata orizzontale (0 <= x <= 319)
y : coordinata verticale (0 <= y <= 199)
ty: una variabile numerica
  1: verifica la presenza di un punto
  0: verifica l'assenza di un punto

Utilizzazione: il comando point viene usato per verificare se un punto è o no presente sullo schermo a determinate coordinate. Ciò è molto utile per gestire le intersezioni.

Esempio:

```
05 rem **********
10 graphic
20 a=0 : d=100 : e=50 : pi=3.14159 : pr=.01 : dim a (100)
25 move a,a : rem *** cursore nell'origine (angolo a sinistra in basso) ***
30 draw a,d,1 : draw d,d,1 : draw d,a,1 : draw a,a,1
40 for i=0 to 100
50 point 25, i, a(i)
60 next i
80 carget" " : rem *** premere qualunque tasto per ripartire ***
90 text
100 for i=0 to 100 : print a(i) : next i
110 rem a(0)+1 : a(100)=1 : linee verticali della maschera
```

Note: nessuna.

display

Funzione: display visualizza una qualunque stringa di caratteri su schermo grafico usando le convenzioni del generatore di schermo su una matrice di 40 colonne e 25 righe (l,c).

Sintassi: display a$,l,c

dove
a$: una qualunque stringa di caratteri
l : numero della riga del punto di inizio (1 <= l <= 25)
c : numero della colonna del punto di inizio (1 <= c <=40)

Utilizzazione: il comando display è molto potente permettendo di unire grafica ad alta risoluzione e caratteri sullo stesso schermo.
Viene usato esattamente come l'istruzione out (vedere pag. 3 - 6).

Esempio:

```
05 rem *** alcuni degli esempi precedenti + le linee 70 e 75 ***
10 graphic
20 a=0 : d=100 : e=50 : pi=3.14159 : pr=.01
25 move a,a : rem *** cursore nell'origine (angolo in basso a sinistra) ***
30 draw a,d,1 : draw d,d,1 : draw d,a,1 : draw a,a,1
40 for i=0 to 2*pi step pr
50 plot e*(1+cos(i)), e*(1+sin(i)),1
60 next i
70 display "Questo è un cerchio", 5,10
75 display "___________", 6,10
80 carget" " : rem *** premere qualunque tasto per ripartire ***
90 text
```

Note:

text

Funzione: text pone fine al modo Graphic ad alta risoluzione e riporta lo schermo in modo Text per la visualizzazione e l'acquisizione dati.

Sintassi: text senza parametri

Esempio:

```
05 rem *** alcuni degli esempi precedenti + le linee 70 e 75 ***
10 graphic
20 a=0 : d=100 : e=50 : pi=3.14159 : pr=.01
25 move a,a : rem *** cursore nell'origine (angolo in basso a sinistra) ***
30 draw a,d,1 : draw d,d,1 : draw d,a,1 : draw a,a,1
40 for i=0 to 2*pi step pr
50 plot e*(1+cos(i)), e*(1+sin(i)),1
60 next i
70 dispaly "Questo è un cerchio", 5,10
75 display "__________", 6,10
80 carget" " : rem *** premere qualunque tasto per ripartire ***
85 ssave 8, " @ 0 : circle. grf"
90 text
95 sclear
96 tline 40,1,1 : tline 40,25,1 : tcol 25,1,1 : tcol 25,1,40
98 ssave 8, " @ 0 : frame.txt"
100 end

list
```

Note:

1) il modo Text è il modo di default all'accensione del sistema

2) se un'istruzione in modo Graphic viene lanciata in modo Text, viene eseguita come se fosse su schermo grafico, ma il suo effetto viene visualizzato solo quando il computer sarà commutato in modo grafico.

3) se un'istruzione in modo Text viene lanciata in modo Graphics essa viene eseguita in attesa delle istruzioni ssave ed sclear.

color

Funzione: color viene usato per caricare nella memoria grafica una matrice colorata 8 punti x 8 punti.

Sintassi: color [–] l,c,lg,ln,co

dove
[–]{ carattere opzionale
l : numero di riga del punto di partenza (1 <= l <= 25)
c : numero di colonna del punto di partenza (1 <= c <= 40)
lg : altezza (o numero delle righe)
ln : lunghezza (o numero delle colonne)
co : nuovo colore (da 0 a 15 - vedere a pag. 3 - 21)

Utilizzazione: il comando color permette di definire un blocco di colore sullo schermo grafico. Viene usato esattamente come il comando rev (pag. 3 - 7).

Esempio:

```
05 rem *** alcuni degli esempi precedenti + le linee 70 e 75 ***
10 graphic
20 a=0 : d=100 : e=50 : pi=3.14159 : pr=.01
25 move a,a : rem *** cursore nell'origine (angolo in basso a sinistra) ***
30 draw a,d,1 : draw d,d,1 : draw d,a,1 : draw a,a,1
40 for i=0 to 2*pi step pr
50 plot e*(1+cos(i)), e*(1+sin(i)),1
60 next i
70 color –1,1,25,40,7
75 color 1,1,25,40,6
80 carget" " : rem *** premere un tasto qualunque per ripartire ***
90 text
```

Note: lo schermo grafico è costituito da un insieme di punti che sono accesi o spenti. La linea 70 spegne i punti gialli, mentre la linea 75 accende i punti blu.

auto

Funzione: auto numera automaticamente le linee di programma.

Sintassi: auto n

dove
n: incremento (1 <= n <= 255)

Utilizzazione: per ottenere l'autonumerazione delle linee di programma è sufficiente battere auto e premere RETURN. Quindi si inizia a scrivere la prima linea di programma e premendo RETURN, appare il numero della linea successiva.

È però possibile cambiare la numerazione automatica introducendo un numero qualsiasi.

Per arrestare la numerazione automatica basta inserire il comando auto senza parametri. Lo stesso effetto si ottiene battendo RETURN subito dopo il numero di linea.

Esempio:

```
auto 10
10 for i=1 to 10
20
```

Note: il comando auto può essere usato solamente in modo diretto e non da programma.

delete

Funzione: delete permette di cancellare linee di programma

Sintassi:
delete a
delete a−
delete −a
delete a−b

dove a,b: numeri di linea del programma

Utilizzazione: il comando delete permette di sopprimere linee di programma e funziona analogamente al comando list.

1) delete a : cancella la linea a
2) delete a− : cancella le linee a partire da a fino alla fine del programma
3) delete −a : cancella le linee a partire dall'inizio del programma fino alla linea a
4) delete a−b: cancella tutte le linee di programma comprese tra le linee a e b.

Esempio: delete 20

Note:
1) il comando delete senza parametri non viene accettato
2) il comando delete può essere usato sia in modo diretto, che da programma.

renu

Funzione: renu effettua la rienumerazione delle linee del programma.

Sintassi: renu [a[,b[,c]]]

dove a,b,c sono parametri opzionali il cui significato è spiegato nel seguito.

Utilizzazione: il comando renu effettua la rinumerazione delle linee di programma (compresi i comandi goto, gosub, etc).

Vi sono 4 possibilità:
1) renu
rinumera l'intero programma
la prima linea è 100
il passo è 10

2) renu a
rinumera l'intero programma
la prima linea è 100
il passo è a

3) renu a,b
rinumera l'intero programma
la prima linea è a
il passo è b

4) renu a,b,c
rinumera una parte del programma
a partire dalla linea a
la prima diventa b
il passo è c

Esempio: renu 20,10
l'intero programma viene rinumerato con passo 10; la prima linea è numerata 20

Note: renu può essere usato solo in modo diretto (non da programma).

dump

Funzione: dump fornisce l'elenco delle variabili utilizzate in un programma ed i valori loro assegnati.

Sintassi: dump senza parametri

Utilizzazione: dump può essere usato in qualunque momento
- dopo aver lanciato un programma
- dopo aver fermato con stop un programma (che viene ripreso con cont)
- dopo un errore di sintassi

dump non fornisce il contenuto di matrici per evitare sovraccarico dello schermo.

Esempio: a=2 : b=3 : a$="test"

dump

si ottiene

a=2
b=3
a$="test"

Note: dump può essere usato sia in modo diretto che da programma.

error

Funzione: error dà la posizione di un errore su una linea in BASIC.

Sintassi: error senza parametri.

Utilizzazione: quando si verifica un errore in un programma BASIC, si arresta l'esecuzione e viene dato un messaggio di errore. Per localizzare l'errore basta battere:

error

In tal modo viene visualizzata la linea con l'errore stesso viene rappresentato in campo inverso sul video.

Esempio:

```
10 a$=STR $(a
20 end
run
error
```

Note:

1) error può essere usato solo in modo diretto (non da programma).

2) dopo che si è verificato l'errore non bisogna usare nessun altro comando prima di error (nemmeno list)

find

Funzione: find permette di cercare una qualunque stringa di caratteri all'interno di un programma.

Sintassi: find <de> ch <de> [A–B]

dove
<de>: un qualunque carattere usato come delimitatore tranne " ed il carattere che si cerca
ch : i caratteri che si cercano
A–B : (opzionali)
A: numero della riga da cui inizia la ricerca
B: numero della riga a cui termina la ricerca

Utilizzazione: il comando find permette di trovare un qualunque carattere od una qualunque stringa di caratteri. Ciò può essere estremamente utile durante la correzione di errori di sintassi. Basta cercare il nome con l'errore.

Esempio:

```
10 ? "WHITE"
20 ? "RED"
30 ? "BLUE"
40 ? "GLU"
50 END

FIND/"GLU"/ : rem or FIND @ glu @
```

Note: find può essere usato sia in modo diretto, che da programma.

trace
off

Funzione: trace permette di eseguire un programma un passo alla volta.
off ritorna al modo normale

Sintassi: trace senza parametri
off senza parametri

Utilizzazione: quando si usa il modo trace il programma gira una istruzione per volta. La linea in esecuzione viene visualizzata sullo schermo.
– per visualizzare la linea premere il tasto shift ad ogni istruzione
– per passare all'istruzione successiva lasciare lo stesso tasto
– per fermare il modo trace premere contemporaneamente il tasto shift ed il tasto run, facendo attenzione di lasciare per primo shift onde evitare il caricamento di un programma.

Esempio: pensate che sussista un errore alla routine 1000.
Effettuate il trace:

```
100 rem il vostro programma
110 trace
120 gosub 1000
130 off
140 rem il resto del vostro programma
```

Note: trace può essere usato sia in modo diretto che da programma.

GENERALITÀ

In questo capitolo sono state raggruppate tutte quelle funzioni che non hanno effetto diretto sullo schermo. Dette funzioni semplificano i controlli, migliorano alcune prestazioni del BASIC e facilitano la programmazione, in particolare lavorando su stringhe di caratteri.

Le funzioni descritte nel capitolo sono:

- hunt : localizza caratteri all'interno di una stringa
- creatst : crea una stringa di caratteri
- if then else: funzione else
- hcopy : effettua la copia dello schermo su stampante
- joy (n) : variabile BASIC per il joystick

Queste funzioni sono indipendenti l'una dall'altra.

hunt

Funzione: hunt trova la posizione di un carattere (o di una serie di caratteri) in una stampa

Sintassi: hunt [–] ca$, a$, pn

dove
– : simbolo opzionale
senza: ricerca il primo carattere uguale a
con : ricerca il primo carattere diverso da
ca$: stringa da trovare
a$: stringa su cui effettuare la ricerca
pn : puntatore che indica dove inizia la ricerca nella stringa a$

Utilizzazione: il comando hunt permette di controllare operazioni (acquisizione, validazione, ...) e di effettuare statistiche. Il risultato viene posto in zo.

Esempio:

```
a$ = "pratica owen"
hunt "a",a$,1
? zo
2
hunt -"e",a$,1
? zo
1
```

Note: se il carattere cercato non esiste nella stringa, zo=0

creatst

Funzione: creatst permette di creare una stringa di caratteri uguali.

Sintassi: creatst a$, ln [,ca$]

dove
ca$: stringa di un carattere (opzionale)
a$: nome della stringa
ln : lunghezza della stringa che si desidera creare.

Utilizzazione: il comando creatst crea una stringa di ln caratteri (ca$). In assenza di ca$, viene usato chr$ (32). È possibile usare qualunque carattere desiderato.

Esempio: creatst a$,80, " "

si ottiene una stringa di 80 blank

creatst b$,10,"*"

si ottiene una stringa di 10 asterischi

Note: questa istruzione sostituisce il seguente loop in BASIC:

```
for i=1 to ln
a$=a$+ca$
next i
```

if then else

Funzione: if then else è una versione potenziata dell'equivalente BASIC, che permette una miglior programmazione.

Sintassi:

```
10 if <condizione> then <istruzioni 1> : else <istruzione 2>
```

oppure

```
10 if <condizioni> then <istruzioni 1>
20 else <istruzioni 2>
```

Utilizzazione: se la condizione è vera, vengono eseguite le istruzioni 1; se è falsa le istruzioni 2. Sono possibili fino a 16 if then else nidificati.

Esempi:

```
10 if <cl> then <il>
20   if <c2> then <i2>
30     if <c3> then <i3>
40     else <e3>
50   else <e2>
60 else <e1>
```

```
10 a=2
20 a=a+1
30 if a=2 then ? "a=2" : else ? "a=3"
40 end
```

Fate girare questo programma, la prima volta così e la seconda togliendo la riga 20.

Note:

1) il comando if then else può essere usato solo da programma.
2) sono possibili fino a 16 istruzioni if then else nidificate. Ciò significa che il numero degli else deve essere uguale od inferiore al numero degli if then.

hcopy

Funzione: hcopy effettua la copia su stampante di tutto ciò che si trova sullo schermo. Se la stampante permette la grafica ad alta risoluzione viene stamapata anche la parte grafica presente sullo schermo.

Sintassi: hcopy senza parametri

Utilizzazione: si scrive semplicemente hcopy

Esempio: si prende uno degli esempi precedenti e si aggiunge la linea:

110 hcopy

Note: 1) l'uso di questa funzione arresta il clock interno del COMMODORE 64 durante l'esecuzione.

2) L'istruzione poke 701,255 permette di ottenere caratteri espansi a doppia larghezza sulla stampante 1515.
L'istruzione poke 701,0 riporta la stampa in modo normale.

3) L'istruzione poke 702,255 permette di stampare a 9 linee per pollice in modo semigrafico.
L'istruzione poke 702,0 permette infine di stampare a 6 linee per pollice.

joy(1), joy(2)

Funzione: joy(1) non è un'istruzione, ma una funzione BASIC nella quale si possono trovare i valori dati dal joystick.

Sintassi: joy(1) joy(2) senza parametri

Utilizzazione: questa variabile permette di usare uno o due joystick senza dover effettuare in memoria istruzioni di peek, ottenendo quindi una estrema semplicità di programmazione.

Esempio: inserire il joystick nel relativo connettore (la prima porta a sinistra del COMMODORE 64).

```
10 ? joy(1) : goto 10
run
```

```
10 rem *** disegna sullo schermo ad alta risoluzione ***
11 graphic : sclear
12 x=125 : y=199 : ad=128 : plot= x,y,1
20 cd=joy(1) : ad=cd
30 if cd=0 then 20
40 if ad<128 then plot x,y,0
50 if cd>128 then cd=cd-128
60 on cd gosub 110,120,100,140,150,160,100,180,190,200
70 plot x,y,1 : goto 20
100 return
110 y=y+1 : return
120 y=y-1 : return
140 x=x-1 : return
150 x=x-1 : y=y+1 : return
160 x=x-1 : y=y-1 : return
180 x=x+1 : return
190 x=x+1 : y=y+1 : return
200 x=x+1 : y=y : return
```

Note: 1) con le istruzioni grafiche la funzione joy(1) permette una programmazione veramente piacevole. Vedere il programma dimostrativo CAO1 che permette di creare con un joystick un diagramma di flusso in modo semiautomatico.

2) joy(2) corrisponde ad un secondo joystick collegato al secondo connettore presente sul COMMODORE 64. La funzione ha lo stesso uso di joy(1).

GENERALITÀ

THE TOOL semplifica la gestione dell'unità a floppy disk con dei semplici e potenti comandi descritti nel seguito:

@ : fornisce il messaggio di errore del floppy disk
@ $: fornisce il catalogo del floppy disk
@ I : inizializza l'unità a floppy disk
@ V : effettua una raccolta (collect)
@ N : nome del file, identificazione
: effettua la formattazione del floppy disk
@ R : nuovo nome di un file=vecchio nome dello stesso file
: cambia il nome di un file
@ S : nome del file
: cancella un file od un programma
@ C : nuovo nome di un file, vecchio nome dello stesso file
: effettua una copia del file

Tutti i comandi descritti debbono essere sempre seguiti dal tasto RETURN.

Note: per ulteriori dettagli sui comandi descritti e sulla loro sintassi, si rimanda ai manuali "Guida per l'utente" del VIC-1540 e del VIC-1541.

Parte I

Generatore di schermo

Istruzione	**Funzione**	**Pagina**
carget a$ [,l,c]	– effettua un get intelligente di un carattere predefinito	3 - 16
clear l,c,lg,ln	– cancella ln caratteri (punto origine: l,c)	3 - 5
clearz n	– cancella la zona n-esima	3 - 15
decz n,l,c,ln[,ty][,f$]	– dichiara la zona n-esina	3 - 10
inz n,a$	– trasferisce il contenuto della zona n-esina in a$	3 - 13
out a$,l,c	– visualizza a$ (punto origine: l,c)	3 - 6
outz n,a$	– visualizza a$ nella zona n-esina e controlla a$	3 - 14
reqz n	– richiede una risposta nella zona n-esina	3 - 12
rev l,c,lg,ln[,co]	– inverte il modo video in una porzione dello schermo	3 - 7
sclear	– cancella tutto lo schermo	3 - 20
screen sc,br[,cr]	– modifica i colori dello schermo, dei bordi e dei caratteri	3 - 21
scroll l,c,ln,lg,ty	– effettua lo scroll di una parte dello schermo	3 - 8
sload 8,"nome"	– carica una pagina dello schermo in memoria	3 - 19
ssave 8,"nome"[,n][,n2 to n3]	– memorizza su floppy disk una pagina dello schermo relativa alle zone definite	3 - 18
tcol ln,l,c	– traccia una linea verticale lunga ln, di origine l,c	13 - 4
tline ln,l,c	– traccia una linea orizzontale lunga ln, di origine l,c	13 - 3

Variabili riservate	**Significato**
zo	– codice ASCII per il tasto return in modo request
ok	– puntatore di stringa

Variabili non riservate	**Significato**
a$	– stringa
c	– numero di colonna (coordinata x)
ca	– codice ASCII di un carattere
co	– colore (nell'istruzione rev)
f$	– formato di stampa
l	– numero di linea (coordinata y)
lg	– larghezza o altezza di una finestra (zona) o numero delle righe
ln	– lunghezza di una finestra o numero delle colonne
n1,n2,n3,n	– numeri di identificazione delle zone
ty	– tipo di formattazione con il comando decz – tipo di scroll (u,d,l,r) con il comando scroll

Istruzione	**Funzione**	**Pagina**
auto n	– numera automaticamente le linee di programma	5 - 2
creatst a$,ln[,ca$]	– crea una stringa di caratteri	6 - 3
color [–]l,c,lg,ln,co	– simile all'istruzione rev, ma in modo Graphic	4 - 8
delete 11-12	– cancella linee di programma (sintassi simile a list)	5 - 2
display a$,l,c	– visualizza su schermo grafico	4 - 6
draw x,y,ty	– disegna un vettore su schermo grafico	4 - 3
dump	– fornisce l'elenco di tutte le variabili usate in un programma ed i valori ad esse assegnati	5 - 4
error	– localizza un errore in una linea di codice BASIC	5 - 5
find <dele><stringa><dele>	– effettua la ricerca di caratteri in un programma	5 - 6
graphic	– commuta il computer da modo Text a modo Graphic	4 - 1
hcopy	– copia automaticamente lo schermo su stampante	6 - 5
hunt [–],ca$,a$,pn	– effettua la ricerca di un carattere all'interno di una stringa	6 - 2
if <c> then <i1> : else <i2>	– istruzione di programmazione strutturata	6 - 4
move x,y	– sposta il cursore in modo grafico senza disegnare	4 - 2
off	– termina il modo di esecuzione passo-passo di un programma	5 - 7
plot x,y,ty	– disegna (o cancella) un punto sullo schermo grafico ad alta risoluzione	4 - 4
point x,y,ty	– verifica la presenza di un punto sullo schermo grafico ad alta risoluzione	4 - 5
renu 11,12,13	– effettua la rinumerazione delle linee di un programma	5 - 3
sound a1,a2,a3,a4,v	– permette di creare un suono (non ancora implementato)	6 - 6
text	– commuta il computer da modo Graphic a modo Text	4 - 7
trace	– attiva il modo di esecuzione passo-passo di un programma	5 - 7

Funzioni riservate	**Contenuto**
joy(1)	– valore dato dal joystick 1
joy(2)	– valore dato dal joystick 2

Variabili non riservate	**Significato**
a$	– una qualunque stringa di caratteri
br	– colore del bordo
c	– punto origine (numero di colonna)
ca	– valore ASCII
ca$	– carattere da ricercare
co	– colore opzionale
cr	– colore del carattere
<c>	– condizione (es: A=B)
<dele>	– delimitatore (qualunque carattere tranne ")
<i1> <i2>	– istruzioni
l	– punto origine (numero di riga)
lg	– altezza (numero di righe)
ln	– lunghezza (numero di colonne)
11,12,13	– numeri di linea di un programma
n	– incremento per la memorizzazione automatica
pn	– puntatore di posizione per la ricerca di un carattere in una stringa
<stringa>	– caratteri da ricercare in un programma
sc	– colore dello schermo
ty	– variabile numerica (1 disegna - 2 cancella)
x,y	– coordinate di un punto dello schermo

generati da THE TOOL

Messaggio	**Spiegazione**
BAD FORMAT	durante l'esecuzione di un comando decz di tipo p, il formato definito non corrisponde a quello della variabile - problemi di lunghezza (vedi pag. 3 - 10)
NO COLOR	manca il parametro del colore oppure il parametro fornito non è permesso (vedi pag. 3 - 21)
NO ZONE	un'istruzione tipo-z è stata lanciata in una zona non dichiarata (vedi da pag. 3 - 9 a 3 - 15)
OUT OF PAGE	le coordinate del punto di origine, oppure il valore dell'altezza forniti in un'istruzione del generatore di schermo sono valori errati.